WoTアーキテクチャにおけるサービス連携のための一貫制御手法の提案

WoT アーキテクチャにおけるサービス連携のための一貫性制御手法の提案

Distributed consensus platform service integration over WoT Architecture

水野　翔輝
Mizuno Shoki
指導教員　藤井 章博

法政大学大学院理工学研究科応用情報工学専攻修士課程

In recent years, IoT(Internet of Things) is being introduced in various fields. For example, in smart home and agriculture field, IoT provide automation and context aware service that *provides service appropriate for user's situation. However many of these systems use a dedicated application that almost incompatible with other systems and variable devices, it is so difficult to integration with each other and extensions. In this paper, we propose a method to construct a platform that enables various devices to collaborate and cooperate with each other, using WoT(Web of Things) architecture that apply Web technology such as Semantic Web and RESTful API to IoT.*

Keywords: Internet of Things, RESTful architectural style, Semantic Web, Web of Things

１．はじめに

近年,様々な分野で IoT(Internet of Things)が導入されつつある.たとえばスマートホームを意識して,ネットワーク接続・制御を可能とする家電製品が販売されている.また,農業分野でも,センサデバイスから取得できる気温・湿度・土壌の状態をネットワーク越しに管理・閲覧できるシステムが導入されている.このように,様々な分野で IoT が広がりを見せつつある.しかし,これらのシステムは専用のアプリケーションを利用する必要があったり,他社のシステムやデバイスとの互換性がないものも多い.こうした現状から,「システムの孤立化(サイロ化)」が重要な課題の一つとなっている.また,スマートホームや農業に対して IoT を応用する場合,接続するセンサやデバイスの数は非常に多くなり,システムが肥大化することが予想される.この状況に対処するため,データの監視と状況に応じたデバイスへの指示を中央集権的に構築する必要があり,センサネットワークの規模に応じて中央端末の処理能力を拡張していかなければならない.

本研究では,これらの課題に対処することを目的とする.１つ目の課題であるシステムのサイロ化については,W3C(WWW Consortium)にて,Web 技術を IoT に適用させる WoT(Web of Things)規格策定の中で議論が進められている[1].そのため,本研究ではサイロ化の解決案として W3C で提唱されている WoT 規格を適用させたプラットフォームのプロトタイプを構築する.このとき,センサネットワーク上のデータやデバイスに対して REST やセマンティック Web の技術を適用する.これらの技術を適用することにより,各デバイスへのインタフェースが標準化される.また,セマンティック Web 技術を用いることにより,データ表現の標準化を試みる.セマンティック Web では,データ表現に用いられるオントロジーが同一であれば,フォーマットが異なるデータあっても統一的に解釈できる.このため,標準的なオントロジーを用いたデータのプロトタイプを構築・提示することで,センサネットワークに対するデータ表現の標準化に対する一例を示せるものと考える.

2 つ目の課題として,センサネットワークの規模拡大によるネットワーク統括端末の肥大化を挙げた.この解決案として,センサネットワークに参加する各デバイスに分散自律処理を行わせるシステムを提案し,その実装例を示す.ここでいう分散自律処理とは,ネットワーク上に存在するデバイスが,各々の保持するデータに応じて自律的に処理を実行することを指す.本研究では,各デバイス自身が取得したデータのみならず,他のデバイスのデータも考慮して複数のデバイスが協調動作を行うシステムを想定している.このため,本システムにはトランザクションの一貫制御のような同期処理も必要になる.また,分散自律処理の実運用には、柔軟に自律分散処理を適

用するためには,デバイスに対して簡単かつ標準的な自律処理の記述・適用ができる必要がある.本研究では,セマンティック Web とそれらを利用した推論機構で分散自律処理を実現する.これにより,システムのサイロ化を考慮した,各デバイス間での協調動作が可能かつ標準的なセンサネットワーク構築の一例を提示できると考える.

２． セマンティック web でのデバイス情報の定義

多種多様なセンサやアクチュエータなどの各種デバイスに関する情報(e.g.設置場所,機能,etc)をセマンティック Web 技術を用いて定義した.システムごとにフォーマットやラベルを独自に定義し,情報を表現した場合にはシステム間の連携が難しく「システムの孤立化(サイロ化)」に繋がってしまう.そのため,本研究ではデータ表現の標準化に対する試みとして,セマンティック Web を用いて構造化データである RDF として定義することで,各種デバイスに関する情報のデータ表現の統一化を試みた.セマンティック Web は,データ表現に用いられるオントロジーが同一であれば,フォーマットが異なるデータであっても統一的に解釈できるという特徴がある.

W3C の SSN(Semantic Sensor Network)オントロジー[2]や M3 オントロジー[3]など,IoT 分野での利用を目的とした主要なオントロジーを用いて,デバイス情報を RDF で表現した.RDF での表現に用いたオントロジーを表１に,語彙やリソースの意味を表２に示す.

表１．デバイス情報の RDF 表現に用いたオントロジー

オントロジー	接頭辞	ネームスペース
RDF(Resource Description Framework)	rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
RDF Schema	rdfs	http://www.w3.org/2000/01/rdf-schema#
SSN ontology	ssn	http://purl.oclc.org/NET/ssnx/ssn#
M3(Machine-to-Machine Measurement)	m3	http://sensormeasurement.appspot.com/m3#
schema.org	shema	http://schema.org/
Basic Geo (WGS84 lat/long) Vocabulary)	geo	http://www.w3.org/2003/01/geo/wgs84_pos#

表２．デバイス情報の RDF 表現に用いた語彙

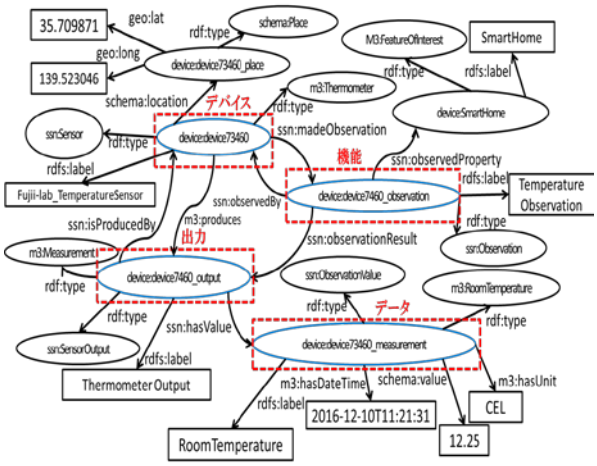
URI	接頭辞表記	意味
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	rdf:type	リソースタイプ
http://www.w3.org/1999/02/22-rdf-syntax-ns#label	rdf:label	リソースラベル
http://www.w3.org/2003/01/geo/wgs84_pos#lat	geo:lat	緯度
http://www.w3.org/2003/01/geo/wgs84_pos#long	geo:long	経度
http://schema.org/location	schema:location	場所
http://schema.org/Place	schema:Place	場所
http://purl.oclc.org/NET/ssnx/ssn#observes	ssn:observes	観測事象
http://purl.oclc.org/NET/ssnx/ssn#FeatureOfInterest	ssn:FeatureOfInterest	特徴
http://purl.oclc.org/NET/ssnx/ssn#observedProperty	ssn: observedProperty	観測事象の特徴
http://sensormeasurement.appspot.com/m3#produces	m3:produces	生成データ
http://sensormeasurement.appspot.com/m3#hasUnit	m3:hasUnit	単位
http://sensormeasurement.appspot.com/m3#hasDateTime	m3:hasDateTime	日時
http://sensormeasurement.appspot.com/m3#hasValue	m3:hasValue	値
http://sensormeasurement.appspot.com/m3#hasName	m3:hasName	名前
http://sensormeasurement.appspot.com/m3#AirThermometer	m3:AirThermometer	空気温度計
http://sensormeasurement.appspot.com/m3#AirThermometer	m3:SmartHome	スマートホーム

表１,表２に示したオントロジーと語彙を用いて,デバイス情報を表現するリソースを定義した.デバイス情報は,①デバイスの名称や設置場所を表現する「デバイスに関する情報」,②センサの観測事象やアクチュエータの機能を表現する「機能に関する情報」,③データの単位や観測時刻を表現する「データに関する情報」,④それら３つのリソースの関連を表現する「出力に関する情報」の４つにリソースに分けて定義した.それぞれのリソースの URI と意味を表３に示す.また,それらのリソース表現を用いて,デバイス情報を RDF で表現したものを図１に示す.

表３．デバイス情報を表現するリソース

URI	接頭辞表記	意味
http://localhost:8080/device/resource/device{デバイスID}}	device:device{デバイスID}	「デバイスに関する情報」を表現するリソース
http://localhost:8080/device/resource/device{デバイスID}}_observation	device:device{デバイスID}_observation	「機能に関する情報」を表現するリソース
http://localhost:8080/device/resource/device{デバイスID}}_measurement{データID}	device:device{デバイスID}_measurement{データID}	「データに関する情報」を表現するリソース
http://localhost:8080/device/resource/device{デバイスID}}_output{出力ID}	device:device{デバイスID}_output{出力ID}	「出力に関する情報」を表現するリソース

上記の表３ようにデバイス情報を複数のリソースを分けて定義することで,より複雑なデバイスの表現や詳細の記述が可能となる.また,表１～表３のオントロジーや①～④として定義した４つのリソースを用いて,デバイス情報を RDF で表現したものを図１に示す.



全体図 注釈付き

図１．デバイス情報を表現した RDF

図１では,主に「デバイス」,デバイスの「機能」,「出力」,「データ」に関する情報を別々のリソースとして定義し,それらを関連付けることでデバイスを表現している.「デバイス」に関する情報とは,デバイス設置場所の緯度・経度などの地理情報や,デバイス名などである.また「機能」では,観測対象事象などの情報を定義しており,「データ」では,単位,値,観測時刻などの情報を定義している.さらに「出力」では「デバイス」,「機能」,「データ」との関係を定義することで,”どのデバイスのどの機能から出力されたデータか”を表現している.また,これらのリソースの関係を全て関連付けることでデバイス情

報を表現している.また,図１のモデルに対して推論を行うことで「device:73460」というURIで示されるセンサは,大気温度という事象を観測するデバイスであり,デバイス名はFujii-lab_TemperatureSensorであることがわかる.本研究では,各デバイス間でデバイスの種類や名前,センサデバイスが取得したデータなどのデバイス情報の共有に上述のRDFを用いた.

３．RESTアーキテクチャによる同期機構

システムを構成する各デバイス上のプロセスで事象(イベント)が発生し,これをシステム全体で共通の順序関係で事象を認識させる方法について述べる.RESTful API を用いる方法では,事象認識の全順序関係を満たすための機構をWebリソースとして提供する.RESTful APIが提供するリソースを表４に示す。

表４．同期APIが提供するリソース

URI	HTTPメソッド	説明
/event/{event_id}/{myid}/{value}	POST	イベントの発生を通知
/receive	GET	イベントの発生状況を取得
/receive/{event_id}	GET	イベントの内容を取得
/ok./{event_id}/{myid}	GET	イベントの認知状況を取得
/ok./{event_id}/{myid}	POST	イベントの認知を通知
/event	DELETE	イベントの削除

表４のURI中の"中かっこ"は変数を表す、それらの変数の意味を表５に示す.

表５．同期APIのURI変数

変数	説明
myid	書くプロセスに割り当てられている一意な番号
event_id	事象を認識するための番号。現在時刻を用いる。
value	イベントの内容

デバイス上の各プロセスは,表４に示したAPIが提供するリソースを利用して,以下の(1)～(6)のいずれかの処理を行う.

(1)イベントの発生を通知

イベントの発生を通知とは,自身のデバイス上のプロセスの状態の変化や,データ更新などが発生した場合に,他の全てのプロセスに対して,その事象(イベント)の発生と内容の通知を行うことを指す.事象の発生通知を行うには,/event/{event_id}/{myid}/{value}というURIで示されるリソースに対してHTTP POSTを行う.このとき,レスポンスとして”Failure”または”Success”が返される.”Failure”は,既に他のプロセスの事象が発生している状態であり,排他的制御により事象の通知が失敗したことを表す.そのため,複数の事象の通知が並列に進行することはない.また,”Success”は事象の通知が成功したことを表す.

(2)イベントの発生状況の取得

イベントの発生状況の取得とは,システム全体で事象が発生しているプロセスの存在の有無,および存在する場合はその事象を一意に識別するIDを取得するものである.事象の発生状況を取得するには”/receive”というURIで示されるリソースに対してHTTP GETを行う.このとき,レスポンスとして”NONE”または”{event_id}”が返される.”NONE”は現在発生中の事象は存在しないことを表す.”{event_id}”は現在発生中の事象のIDを表し,(1)の”事象の発生通知”を行った際に用いた変数である.また,(1)で示したように複数の事象が並列に発生することはない.そのため,複数の”{event_id}”がレスポンスとして返されることはない.

(3)イベントの内容の取得

イベントの内容の取得とは,現在発生している事象の内容を取得するものである.事象の内容を取得するには”/receive/{event_id}”というURIで示されるリソースに対してHTTP GETを行う.このとき,レスポンスとして”Failuer”または”{value}”が返される.ここで返される{value}は,(1)の事象の発生通知を行った際に用いた変数である.また,事象の内容の取得に用いるURIに含む変数{event_id}は,(2)で返されるレスポンスの内容である.

(4)イベントの認知を通知

イベントの認知を通知とは,現在発生している事象の内容取得が完了し,それを他の全てのプロセスに対して通知を行うこと指す.事象の認知を通知するには”/ok/{event_id}/{my_id}”で示されるリソースに対して,HTTP POSTを行う.このとき,レスポンスとして”Success”または”Failure”が返される.レスポンスの”Success”は,通知が完了したことを表す.また,”Failure”はリクエストで指定した{event_id}で示される事象が存在しないことを表す.

(5)イベントの認知状況を取得

イベントの認知状況の取得とは,(4)の”事象の認知を通知”を行ったプロセスの{myid}の一覧を取得するものである.事象の認知状況を取得するには”/ok/{event_id}/”というURIで示されるリソースに対してHTTP GETを行う.このとき,レスポンスとして”Failure”または１つ以上の”{my_id}”を返す.ここで返される”Failure”は,”{event_id}”で示される現在発生中の事象が存在しないことを表す.また,レスポンスに含まれる”{my_id}”は,URIで変数として指定した”{event_id}”で示される事象の発生を認知しているプロセスのIDを表す.

(6)イベントの削除

イベントの削除とは,(1)において”Success”が返された事象に対して,その事象の通知処理を終了するものである.例えば,プロセス全体で現在発生中の事象の認識が完了し,全てのデバイス上のプロセスで更新処理が完了した場合に実行される.事象の削除を行うには”/event”に対してHTTP DELETEを行う.このとき,(1)の”事象の発生通知”を実行し,事象を発生させたプロセスが”事象の削除”を行うものとする.

システムを構成する各サブシステム上の各プロセスは,自身の状態及び(1)～(6)の結果から,自身の状態を遷移させる.これを繰り返すことで各プロセスは共通の順序で事象を認識する.各デバイスの状態遷移を図２に示す.

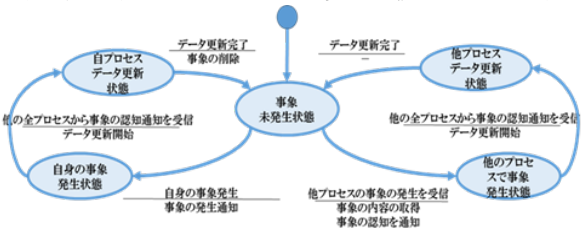


図４．デバイスの状態遷移

各プロセスは図４のように,状態を遷移させることで全順序関係を満たした事象の認識を行う.また(1)で述べたようにシステム全体で複数の事象の通知処理が並列に進行することはなく,他の全てのプロセスが事象を認知するのを待って次の事象の通知処理に移る.これにより,全てのプロセスで共通の順序で事象の認識を行う.

４．実装

デバイス自身が取得したデータのみならず,他のデバイスのデータも考慮して複数のデバイスと協調動作が行えるプラットフォームを構築した.また,ユーザからのデバイス情報の利用や,他の Web アプリケーションとの連携が行えるユーザインタフェースも併せて実装した.デバイス端末は,小型コンピュータボード Raspberry pi を採用した.またサーバは,OS に Ubuntu Linux,アプリケーションコンテナとして Tomcat,RDF データベースに Virtuoso を用いて機能を実装した.図３にシステム構成図を示す.本システムは,各デバイスの持つデータを共有するための機能である「デバイス同期・制御機能」,ユーザや Web アプリケーションに対してデバイス情報を提供する「Web サービス機能」からなる.

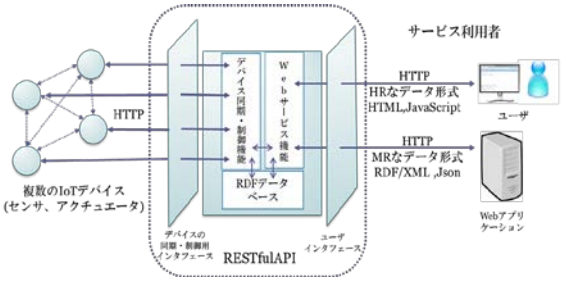


図 ３　システムの構成図

(I)デバイスの同期・制御機能

デバイスの同期･制御機能は,各デバイス間の状態やデータの共有を行う機能を提供する.デバイスの状態や観測データは,RDF として各デバイスに格納され,本機能を用いて全てのデバイスで共有される.前述した REST アーキテクチャによる同期機構の実装である.

(II)Web サービス機能

Web サービス機能では,ユーザや Web アプリケーションに対して,各デバイスの設置場所や機能,デバイスが保持するデータなどの情報を提供する.各デバイスに関するデータは,ユーザと Web アプリケーションは,ユーザインタフェースに対して HTTP リクエストを送信することで,各種デバイス情報の参照やデータ取得が行える.表６にユーザインタフェースの URI と意味を示す.

表６．ユーザインタフェース

URI	説明
/device/resource	全デバイスに関するインデックス
/device/resource/device{デバイスID}}	デバイスの設置場所、名前
/device/resource/device{デバイスID}}_observation	デバイスの機能詳細
/device/resource/device/{デバイスID}_output	観測データなどのインデックス
/device/resource/device{デバイスID}}_measurement{データ番号}	値、単位、観測日時などのデータ詳細

ユーザインタフェースでは,ユーザに対しては Web ブラウザを通して人間が理解しやすい情報形式(HR:Human Readable)であるページとして情報提供する.またそれに対して,Web アプリケーションに対しては,機械が理解可能(MR:Machine Readable)な JSON-LD や RDF/XML,CSV 形式での情報提供を行う.MR/HR なデータ形式の判別は,HTTP の Accept ヘッダにて行う.

５．実行結果

(a)デバイスのデータ同期

温度センサを実装した３台のデバイスにて,各デバイスの種類や観測データの同期を行わせた結果,あるデバイスの観測データの更新が発生した場合に,全てのデバイスで共通の順序でデータが更新されることを確認した.デバイスの動作管理画面を以下の図４に示す.図内の各枠は,接続されている各デバイスの種類や,観測データの内容を表している.また,色付きの枠はデバイス情報の更新があったことを示している.

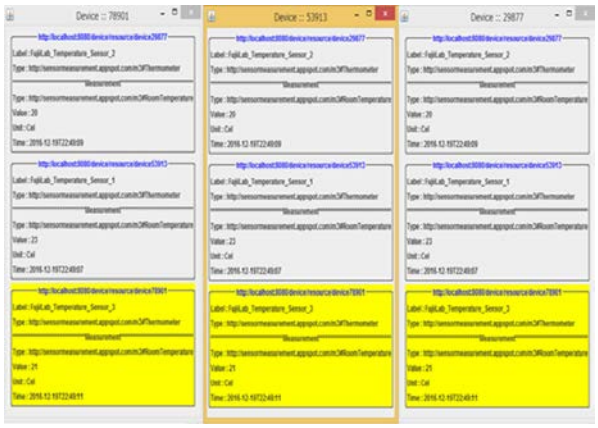


図４．デバイスの動作管理画面

(b)ユーザインタフェース

Web ブラウザを利用して,サーバ上のユーザインタフェースにリクエストを送信し,デバイスの種類や観測データなどの情報がページとしての表示されることを確認した.表示画面を図５に示す.デバイスの設置場所は,Google Maps を利用した地図上にマッピングされ,観測データのグラフとデバイスの種類などの情報はポップアップウィンドウ内に表示される.

図５．ユーザインタフェース

６．考察・結論

IoTの普及に伴い,デバイスをWeb上で管理運用するためのアーキテクチャが求められている事から[1],Web 技術を用いたシステムの構築を行った.構築したシステムのサービスの連携性については,セマンティック Web 技術を用いてデバイス情報を表現することで,他のシステムとのデータ表現の統一化を図る一例を示した.セマンティック Web を IoT 分野へ応用を行うプロジェクトは複数存在する.しかし,現在,利用可能なサービスとして公開されているものは確認できなかった.そのため､本研究では他のシステムとのサービス連携についての実検証には至っていない.今後は､IoT 分野や W3C で策定中である WoT 仕様に関する動向の調査を行った上で,他のシステムとの連携性などの詳細な検証を行っていきたい.また,REST アーキテクチャによる同期機構については,HTTP や RESTful Web API などの代表的な Web 技術を用いているため,既存の Web アプリケーションとの親和性が高い.そのため,複数のデバイス間だけではなく,Web アプリケーションを含めた同期による協調動作が可能である.したがって,本機構を利用することで,Web とデバイスそれぞれの利点を生かしたサービスの構築ができると考える.本システムの実利用については,農業分野などの圃場管理や,スマートホーム分野などでのホームオートメーションシステムなどの具体的なサービスにおいて,一定時間の運用事例の蓄積を行う必要があると考える.

謝辞：本研究を進めるにあたり、熱心なご指導を頂きました藤井章博教授、研究活動を支えて下さった法政大学大学院 工学研究科 平成 26 年度 修士課程修了 清水宏康氏、藤井研究室の皆様に感謝致します。

参考文献

[1]W3C, “WEB OF THINGS ”, https://www.w3.org/WoT/, 2017/2/15 閲覧
[2]藤井章博, 中村眞, 根本義章,”連続同期を効率良く処理する分散型同期制御方式”,電子情報通信学会論文誌, Vol.J81-D-I, No.3, pp.244-252
[3]W3C,”OWL-Semantic Web Standards”https://www.w3.org/OWL/,2017/2/15 閲覧
[4]W3C Semantic Sensor Network Incubator Group(SSN- XG), “Semantic Sensor Network Ontology”, http://www.w3.org/2005/Incubator/ssn/ssnx/ssn, 2017/2/15 閲覧
[5]schema.org, schema.org, http://schema.org ,2017/2/15 閲覧
[6]Gyrard et al, Enrich machine-to-machine data with semantic web technologies for cross-domain applications, WF-IOT 2014
[7]清水宏泰、藤井章博「Semantic REST による RDF データの活用」情報処理学会論文誌、第 55 巻、2 号、2014 年
[8]A.Fujii, H. Shimizu,” RDFa Parser for Hybrid Resources”,Semantic Technology LNCS8943、Springer (2015)
[9]江上周作、川村隆浩、藤井章博、大須賀明彦、「BOM エージェントの実現に向けた LOD の構築」、電子情報通信学会論文誌 D Vol. J 98 D/ pp.992-1004 (2015)
[10]Cisco Systems, The Internet of Things How the Next Evolution of the Internet Is Changing Everything,http://www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf,2017/2/15 閲覧
[11]長野伸一, Linked Data とセンサネットワーク, 人工知能学会誌 27(2), 200-206, 2012-03-01
[12]神崎正英 , セマンティック・ウェブのための RDF/OWL 入門, 森北出版, 2005.